Progress (100%)

Item 1/30

The following code:

```
print(float("1.3"))
```

- (●) prints `1.3`
- ( ) prints `13`
- ( ) prints `1,3`
- ( ) raises a `ValueError` exception

Next →

Retake Test | Back to Summary

Item 2/30

The following statement:

```
assert var != 0
```

- has no effect
- will stop the program when `var == 0`
- is erroneous
- **will stop the program when `var != 0`**

Prev Next

Retake Test Back to Summary

Item 3/30

What will be the output of the following snippet?

```
try:
    raise Exception
except:
    print("c")
except BaseException:
    print("a")
except Exception:
    print("b")
```

- it will cause an error
- `b`
- `a`
- `1`
- `1`

Prev Next

Retake Test Back to Summary

Item 4/30

What will be the result of executing the following code?

```
try:
    raise Exception(1, 2, 3)
except Exception as e:
    print(len(e.args))
```

- it will print 2
- **it will print 3**
- it will raise an unhandled exception
- it will print 1

Prev Next

Retake Test Back to Summary

Item 5/30

If `s` is a stream opened in *read* mode, the following line:

```
q = s.read(1)
```

will read:

- (•) one character from the stream
- ( ) one buffer from the stream
- ( ) one line from the stream
- ( ) one kilobyte from the stream

Prev Next

Retake Test Back to Summary

Item 6/30

What will be the result of executing the following code?

```
class A:
    def __init__(self):
        pass

a = A(1)
print(hasattr(a, 'A'))
```

- it will print `True`
- it will print `False`
- **it will raise an exception**
- it will print `1`
- it will print `1`

← Prev Next →

Retake Test Back to Summary

Item 7/30

What will be the result of executing the following code?

```
def o(p):
    def q():
        return '*' * p
    return q

r = o(1)
s = o(2)
print(r() + s())
```

- it will print `*`
- **it will print `***`**
- it will print `**`
- it will print `****`
- it will print `****`

← Prev Next →

Retake Test Back to Summary

Item 8/30

The following statement:

```
from a.b import c
```

causes the import of:

- (●) entity `c` from module `b` from package `a`
- ( ) entity `c` from module `a` from package `b`
- ( ) entity `b` from module `a` from package `c`
- ( ) entity `a` from module `b` from package `c`

← Prev Next →

Retake Test Back to Summary

Item 9/30

What will be the output of the following snippet?

```
try:
    raise Exception
except BaseException:
    print("a")
except Exception:
    print("b")
except:
    print("c")
```

- a
- b
- 1
- it will cause an error

Prev Next

Retake Test Back to Summary

Item 10/30

The `sys.stderr` stream is normally associated with:

- a null device
- the screen
- the printer
- the keyboard

Prev Next

Retake Test Back to Summary

Item 11/30

What will be the result of executing the following code?

```
class A:
    pass

class B(A):
    pass

class C(B):
    pass

print(issubclass(A, C))
```

- it will raise an exception
- it will print `1`
- it will print `False`
- it will print `True`
- it will print `True`

← Prev Next →

Retake Test Back to Summary

Item 12/30

What will be the result of executing the following code?

```
def I(n):
    s = '+'
    for i in range(n):
        s += s
        yield s

for x in I(2):
    print(x, end='')
```

- it will print `++++++`
- it will print `+`
- it will print `+++`
- it will print `++`

Prev Next

Retake Test Back to Summary

Item 13/30

If you want to fill a byte array with data read in from a stream, you can use:

- the `readbytes()` method
- the `readfrom()` method
- the `readinto()` method
- the `read()` method
- the `read()` method

Prev Next

Retake Test Back to Summary

Item 14/30

What will be the result of executing the following code?

```
class A:
    def a(self):
        print('a')

class B:
    def a(self):
        print('b')


class C(B, A):
    def c(self):
        self.a()

o = C()
o.c()
```

- it will raise an exception
- it will print `a`
- it will print `c`
- it will print `b`

Prev Next

Retake Test Back to Summary

Item 15/30

What will be the result of executing the following code?

```
class I:
    def __init__(self):
        self.s = 'abc'
        self.i = 0
    def __iter__(self):
        return self
    def __next__(self):
        if self.i == len(self.s):
                raise StopIteration
        v = self.s[self.i]
        self.i += 1
        return v

for x in I():
    print(x, end='')
```

- it will print `210`
- it will print `cba`
- **it will print `abc`**
- it will print `012`
- it will print `012`

Prev Next

Retake Test Back to Summary

Item 16/30

What will be the output of the following code, located in the `p.py` file?

```
print(__name__)
```

- `__p.py__`
- `p.py`
- `main`
- `__main__` ✓

← Prev Next →

Retake Test Back to Summary

Item 17/30

The following code:

```
x = "\\\\"
print(len(x))
```

- prints 3
- will cause an error
- prints 1
- prints 2

← Prev Next →

Retake Test Back to Summary

Item 18/30

What will be the effect of running the following code?

```
class A:
    def __init__(self, v):
        self.__a = v + 1

a = A(0)
print(a.__a)
```

- it will raise an `AttributeError` exception
- it will print `2`
- it will print `1`
- it will print `0`

← Prev Next →

Retake Test Back to Summary

Item 19/30

Knowing that a function named `fun()` resides in a module named `mod`, and was imported using the following statement:

```
from mod import fun
```

Choose the right way to invoke the `fun()` function:

- `mod:fun()`
- **`fun()`**
- `mod.fun()`
- `mod::fun()`
- `mod::fun()`

← Prev Next →

Retake Test Back to Summary

Item 20/30

The following line of code:

```
for line in open('text.txt', 'rt'):
```

- may be valid if `line` is a list
- **is valid as `open` returns an iterable object**
- is invalid as `open` returns nothing
- is invalid as `open` returns a non-iterable object

← Prev Next →

Retake Test Back to Summary

Item 21/30

What will be the output of the following code?

```
class A:
    A = 1
    def __init__(self):
        self.a = 0

print(hasattr(A, 'a'))
```

- 1
- **False**
- 0
- True

← Prev Next →

Retake Test Back to Summary

Item 22/30

The following code prints:

```
print(chr(ord('p') + 2))
```

- t
- q
- r
- s
- 3

Prev Next

Retake Test Back to Summary

Item 23/30

The following code:

```
x = "\\\"
print(len(x))
```

- prints 1
- prints 2
- prints 3
- will cause an error

Prev Next

Retake Test Back to Summary

Item 24/30

If the class's constructor is declared as below, which one of the assignments is invalid?

```
class Class:
    def __init__(self, val=0):
        pass
```

- `object = Class(None)`
- **`object = Class(1, 2)`**
- `object = Class(1)`
- `object = Class()`

`object = Class()`

← Prev Next →

Retake Test Back to Summary

Item 25/30

What output will appear after running the following snippet?

```
import math
print(dir(math))
```

- the number of all the entities residing in the `math` module
- **a list of all the entities residing in the `math` module**
- an error message
- a string containing the fully qualified name of the module

← Prev Next →

Retake Test Back to Summary

Item 26/30

Assuming that the following three files: `a.py`, `b.py`, and `c.py` reside in the same folder, what will be the output produced after running the `c.py` file?

```
# file a.py
print("a", end='')

#file b.py
import a
print("b", end='')

#file c.py
print("c", end='')
import a
import b
```

- `abc`
- `bac`
- `cba`
- `cab`

← Prev Next →

Retake Test Back to Summary

Item 27/30

What will be the output of the following code?

```
class A:
    def __init__(self, v=2):
        self.v = v

    def set(self, v=1):
        self.v += v
        return self.v

a = A()
b = a
b.set()
print(a.v)
```

- 3
- 1
- 0
- 2

Prev Next

Retake Test Back to Summary

Progress (100%)

Item 28/30

The compiled Python bytecode is stored in files which have their names ending with:

- `py`
- `pc`
- `pyc`
- `pyb`

Prev Next

Retake Test Back to Summary

Item 29/30

Assuming that the `open()` invocation has gone successfully, the following snippet will:

```
for x in open('file', 'rt'):
    print(x)
```

- read the file character by character
- read the whole file at once
- **read the file line by line**
- cause an exception

Prev Next

Retake Test Back to Summary

Item 30/30

If there are more than one `except:` branches after the `try:` clause, we can say that:

- one or more of the `except:` blocks will be executed
- exactly one of the `except:` blocks will be executed
- none of the `except:` blocks will be executed
- **not more than one `except:` block will be executed**

Prev

Retake Test

Back to Summary